

Задача A22. Циклы

 1 с  256 МБ

Ян — храбрый мальчик, который всегда идёт кратчайшим путём из школы к дому своей бабушки. К сожалению, этот путь пролегает через зачарованный лес.

Однажды, проходя через лес, Ян встречает ужасное существо Тунг Тунг Саур. Существо предлагает ему сыграть в игру: если Ян выиграет, ему разрешат продолжить путь — и, возможно, даже дадут награду. Если же он проиграет, Тунг Тунг Саур съест его.

Игра начинается с того, что Тунг Тунг Саур выбирает скрытую перестановку P из N элементов. Теперь очередь Яна. Он может выбирать две разные позиции в перестановке и просить Тунг Тунг Саура поменять местами элементы на этих позициях. Тунг Тунг Саур выполнит обмен и объявит количество **циклов**¹ в обновлённой перестановке. Результат обменов сохраняется — Тунг Тунг Саур не отменяет их перед следующим запросом (но Ян может повторить обмен, чтобы отменить предыдущий, если хочет).

В любой момент Ян может объявить, что перестановка отсортирована, крикнув “Sorted”. Если перестановка действительно отсортирована по возрастанию, Тунг Тунг Саур отпускает его. Более того, Тунг Тунг Саур наградит Яна золотыми монетами, количество которых зависит от того, сколько обменов потребовалось. Поэтому Ян должен попытаться отсортировать перестановку, используя как можно меньше обменов.

Ваша задача — помочь Яну отсортировать скрытую перестановку, используя только предоставленную информацию, и постараться сделать это с минимальным количеством обменов.

Детали реализации

Вам необходимо реализовать функцию:

```
void sortPermutation(int N);
```

Она будет вызвана один раз для каждого теста с параметром N — количеством элементов в перестановке, которую нужно отсортировать. Из этой функции (и других ваших функций) вы можете вызывать функцию `performSwap` для обмена двух элементов перестановки, и в ответ получите общее количество циклов в получившейся перестановке. Вы должны гарантировать, что после завершения вашей функции скрытая перестановка будет отсортирована по возрастанию: $P_i = i$ для всех $0 \leq i < N$, так как Ян сразу крикнет “Sorted” после выхода из этой функции.

```
int performSwap(int x, int y)
```

Вызов `performSwap` представляет обмен элементов P_x и P_y . Обратите внимание, что если вы вызовете эту функцию с одинаковыми элементами, т.е. $x = y$,

¹ Любую перестановку целых чисел $\{0, 1, \dots, N-1\}$ можно разложить на набор непересекающихся циклов. Чтобы найти такой цикл, начните с любого индекса i и следуйте по отображению $i \mapsto P_i \mapsto P_{P_i} \mapsto \dots$, пока не вернётесь к i — это и будет один цикл. Например, перестановка $[1, 2, 0, 5, 4, 3]$ представляет отображение $0 \mapsto 1, 1 \mapsto 2, 2 \mapsto 0, 3 \mapsto 5, 4 \mapsto 4$ и $5 \mapsto 3$, что даёт непересекающиеся циклы $(0\ 1\ 2)$, $(3\ 5)$ и (4) , то есть всего 3 цикла.

XVI МЕЖДУНАРОДНЫЙ ТУРНИР ПО ИНФОРМАТИКЕ ШУМЕН 2025

вы получите вердикт Wrong Answer. Функция принимает номера в 0-индексации x и y элементов для обмена и возвращает количество циклов в **обновлённой** перестановке.

Локальное тестирование

Для локального тестирования предоставляются локальный грейдер и заголовочный файл. Локальный грейдер сначала считывает N , а затем N различных чисел от 0 до $N - 1$. После этого он вызывает вашу функцию `sortPermutation` и ожидает, что она корректно отсортирует перестановку.

Ограничения

- $N = 1000$

Подзадачи

Подзадача	Баллы	Ограничения	Дополнительно
1	10	$N = 1000$	Для чётных i : $(P_i, P_{i+1}) = (i, i + 1)$ или $(i + 1, i)$.
2	20	$N = 1000$	Загаданную перестановку можно отсортировать, используя ровно один обмен.
3	70	$N = 1000$	–

Вы получаете долю баллов за данную подзадачу, только если корректно пройдены все тесты в её составе. Ваш результат будет вычислен на основе Q — максимального количества вызовов функции `performSwap` на этих тестах, и рассчитан по следующей формуле:

$10^7 < Q$	0%
$10^6 < Q \leq 10^7$	10%
$9 \cdot 10^4 < Q \leq 10^6$	20%
$3 \cdot 10^4 < Q \leq 9 \cdot 10^4$	60%
$Q \leq 3 \cdot 10^4$	100%

Пример теста

Ввод	Взаимодействие
3 2 0 1	<code>sortPermutation(3)</code> <code>performSwap(0, 1): return 2</code> <code>performSwap(0, 1): return 1</code> <code>performSwap(0, 1): return 2</code> <code>performSwap(1, 2): return 3</code> <code>sortPermutation(3): returns</code>